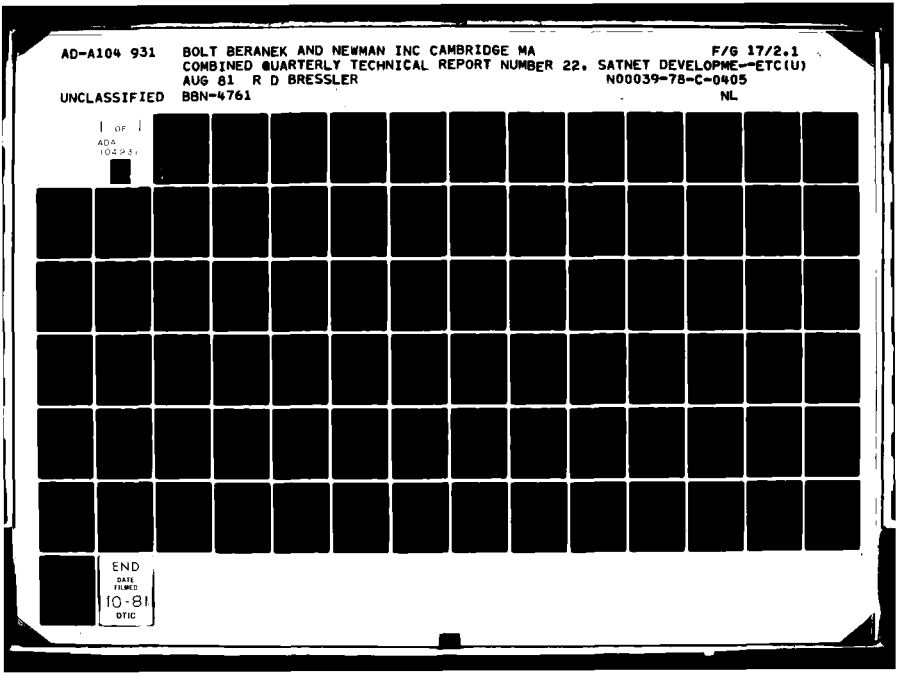
AD-A104 931
BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA
F/G 17/2.1
COMBINED QUARTERLY TECHNICAL REPORT NUMBER 22. SATNET DEVELOPME--ETC(U)
AUG 81 R D BRESSLER
N00039-78-C-0405
UNCLASSIFIED
BBN-4761
NL
OF
ADA
104931
END
DATE
FILMED
10-81
DTIC

AD A104931

Bolt Beranek and Newman Inc.

Report No. 4761

LEVEL II

Combined Quarterly Technical Report No. 22

SATNET Development and Operation
Pluribus Satellite IMP Development
Remote Site Maintenance
Internet Development
Mobile Access Terminal Network
TCP for the HP3000
TCP-TAC
TCP for VAX-UNIX

August 1981

Prepared for:
Defense Advanced Research Projects Agency

DTIC
ELECTE
OCT 1 1981
D

DTIC FILE COPY

DISTRIBUTION STATEMENT
Approved for public release;
Distribution Unlimited

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A104931	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) COMBINED QUARTERLY TECHNICAL REPORT No. 22		5. TYPE OF REPORT & PERIOD COVERED 5/1/81 to 7/30/81
		6. PERFORMING ORG. REPORT NUMBER 4761
7. AUTHOR(s) R. D. Bressler		8. CONTRACT OR GRANT NUMBER(s) MDA903-80-C-0353 & 0214 N00039-78-C-0405 N00039-79-C-0386 N00039-80-C-0664
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 10 Moulton Street Cambridge, MA 02238		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS ARPA Order Nos. 3214 and 3175.17
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209		12. REPORT DATE August 1981
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) DSSW Rm. 1D, The Pentagon Washington, DC 20310 NAVELEX Washington, DC 20360		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE/DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer networks, packets, packet broadcast, satellite communication, gateways, Transmission Control Program, UNIX, Pluribus Satellite IMP, Remote Site Module, Remote Site Maintenance, shipboard communications, Terminal Access Controller, VAX.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This Quarterly Technical Report describes work on the development of and experimentation with packet broadcast by satellite; on development of Pluribus Satellite IMPs; on a study of the technology of Remote Site Maintenance; on the development of Inter-network monitoring; on shipboard satellite communications; and on the development of Transmission control protocols for the HP3000, TAC, and VAX-UNIX.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Report No. 4761

COMBINED QUARTERLY TECHNICAL REPORT NO. 22

SATNET DEVELOPMENT AND OPERATION
PLURIBUS SATELLITE IMP DEVELOPMENT
REMOTE SITE MAINTENANCE
INTERNET DEVELOPMENT
MOBILE ACCESS TERMINAL NETWORK
TCP FOR THE HP3000
TCP-TAC
TCP FOR VAX-UNIX

August 1981

This research was supported by the Defense Advanced Research Projects Agency under the following contracts:

N00039-78-C-0405, ARPA Order No. 3175.17
MDA903-80-C-0353. ARPA Order No. 3214
MDA903-80-C-0214, ARPA Order No. 3214
N00039-80-C-0664
N00039-81-C-0408

Submitted to:

Director
Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington. VA 22209

Attention: Program Management

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies. either expressed or implied. of the Defense Advanced Research Projects Agency or the U.S. Government.

Table of Contents

Accession For
NTIS GRA&I
DTIC TAB
Unannounced
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special

1 INTRODUCTION

This Quarterly Technical Report is the current edition in a series of reports which describe the work being performed at BBN in fulfillment of several ARPA work statements. This QTR covers work on several ARPA-sponsored projects including (1) development and operation of the SATNET satellite network; (2) development of the Pluribus Satellite IMP; (3) Remote Site Maintenance activities; (4) inter-network monitoring; (5) development of the Mobile Access Terminal Network; (6) TCP for the HP3000; (7) TCP-TAC; and (8) TCP for the VAX-UNIX. This work is described in this single Quarterly Technical Report with the permission of the Defense Advanced Research Projects Agency. Some of this work is a continuation of efforts previously reported on under contracts DAHC15-69-C-0179, F08606-73-C-0027. F08606-75-C-0032, MDA903-76-C-0213. MDA903-76-C-0252, and N00039-79-C-0386.

2 SATNET DEVELOPMENT AND OPERATION

As part of our participation in the Atlantic Packet Satellite Experiment (SATNET) during the last quarter, we have been reworking the Satellite IMP software to accommodate a network with more than four Satellite IMPs, a prerequisite for adding new sites to SATNET. This requires changing the algorithm for transmitting Hello packets, modifying the Hello report format of the TENEX programs RECORDER and MONITOR, implementing a second status word for display of channel reception on the Honeywell 316 console lights, adding new host addresses, and changing the mechanism for handling group addresses. Satellite IMP Version 3.4:1, released in July, is this software but limited to the current four Satellite IMPs in the field.

We also added the capability of printing traps as they occur on the on-site monitor terminal. These traps, which have previously been sent to RECORDER only, will aid in on-site debugging at remote sites. The format of the trap printout is:

TIME GLOBAL-TIME ** TRAP# (A) (X)

where TIME is a 16-bit word in units of seconds representing the interval since the Satellite IMP last restarted. GLOBAL-TIME is a 16-bit word in units of Virtual Slots (10.24 milliseconds) representing the commonly agreed satellite time, TRAP# is the

trap identification number, (A) is the contents of the A register. and (X) is the contents of the X register. Although GLOBAL-TIME is the fundamental clock in all Satellite IMPs. the interval since the Satellite IMP last restarted is included to resolve ambiguity due to GLOBAL-TIME being kept modulo 11 minutes (more precisely 671.09 seconds).

For maintainability and ease of use, all configuration-dependent tables and constants in the Satellite IMP software (e.g., pathways to TENEX, default parameters. etc.) have been moved into a separate source file. Since MATNET and SATNET share most of the remaining files. software modifications are automatically incorporated into both networks.

2.1 Hello Packet Transmission

Currently, a constant bandwidth is assigned to the transmission of Hello packets by each Satellite IMP every 128 Virtual Slots or equivalently 1.31 seconds. This interval is thereby designated as a Hello frame. In preparation for the transition to a network with more than four Satellite IMPs. the ordering of Hello packet transmissions is now fixed rather than rotated during successive frames. Were the rotated ordering kept. incommensurability between the number of sites and the

number of Hello frames in the reporting interval would unduly complicate the MONITOR display of Hello packet reception and the Satellite IMP algorithms processing Hello packets upon reception and determining when to send Hello packets. Since a rotating Hello pattern was implemented originally to reveal AGC problems with the modified SPADE modems. the presence at all sites of the Linkabit digital modems with elaborate T&M capabilities has made this feature unnecessary. As an additional benefit, site personnel, being able to identify transmissions from specific sites unambiguously by the position of the Hello packet in the frame. will now be able to adjust relative powers more easily.

Since every 64 Hello frames all Satellite IMPs synchronously report to RECORDER on how well they have heard Hello packet transmissions, versions of RECORDER and MONITOR that understand the new Hello packet ordering were released concurrently with the new Satellite IMP software. The new MONITOR Hello report format is presented below along with a sample printout to clarify its use. (Characteristic of all MONITOR reports is an almost cryptic succinctness resulting from the necessity of presenting large quantities of information in a small space.) For completeness, we also present here the other MONITOR report formats.

HELLO REPORT FORMAT

s time Hello = (ck) e. g. t. c

where:

s -- Reporting site identifier (E=Etam, G=Goonhilly, T=Tanum, C=Clarksburg);

time -- Local TENEX time;

ck -- Total Hello packets received in the past 64 frames which have a bad hardware checksum and a good software checksum;

e -- Total of Etam's Hello packets missed in the past 64 frames;

g -- Total of Goonhilly's Hello packets missed in the past 64 frames;

t -- Total of Tanum's Hello packets missed in the past 64 frames;

c -- Total of Clarksburg's Hello packets missed in the past 64 frames.

As an example, the following report from Etam at 1200 local TENEX time shows 3 checksum errors. 0 missing Hello packets from Etam and Goonhilly, 11 missing packets from Tanum, and 64 missing packets from Clarksburg. (The latter indicates Clarksburg was not participating in the network.)

E 1200 Hello = (3) 0, 0, 11, 64

PROGRAM STATUS REPORT FORMAT

s time (vers) prot STR=a R=b BCK=c {A1 A2 A3 M C N loops}

where:

s -- Reporting site identifier {E=Etam, G=Goonhilly, T=Tanum, C=Clarksburg};
time -- Local TENEX time;
vers -- Satellite IMP version number;
prot -- Current channel protocol {CPODA. FPODA. FTDMA};
a -- Total number of streams in existence;
b -- Nominal receive rate in Kb/s {64, 32, 16};
c -- Average number of background passes per Virtual Slot;

displayed switches:

A1 -- First access control parameter is on (preventing channel access);
A2 -- Second access control parameter is on (preventing channel access);
A3 -- Third access control parameter is on (preventing channel access);
M -- One or more message generator components are on;
C -- Cumstats are being sent to a host;
N -- One or more noise gates are on;
0I -- Satellite channel looped internally;
0X -- Satellite channel looped externally via a signal from the Satellite IMP;
0L -- Satellite channel looped. but not by Satellite IMP;
1I -- Line to host 1 looped internally
1X -- Line to host 1 looped externally via a signal from the Satellite IMP;
1L -- Line to host 1 looped, but not by Satellite IMP;
2I.2X,2L same as 1I.1X,1L but for host 2.

CHANNEL TRAFFIC REPORT FORMAT

s time E=a/b/c S=d*e/f/g*h C=i/j/k/lc/ls DS=m/n/p DR=q/r/s

where:

s -- Reporting site identifier (E=Etam, G=Goonhilly. T=Tanum, C=Clarksburg);
time -- Local TENEX time;

a -- Total packets received with bad hardware checksum and bad software header checksum;
b -- Total packets received with bad hardware checksum and good software header checksum;
c -- Total Hello packets received;

d -- Total Hello frames with site out of frame sync;
e -- Total PODA frames with site in datagram in-sync res-sync state;
f -- Total PODA frames with site in datagram out-of-sync res-sync state;
g -- Total PODA frames with site in datagram initial acquisition res-sync state;
h -- Total PODA frames with site in stream sending-for-Help state;

i -- Total control packets heard from other sites;
j -- Total control packets heard from this site;
k -- Total control packets sent;
lc -- Fraction of total bandwidth (including hello subframe) used for control subframes;
ls -- Fraction of total bandwidth used for streams;

m -- Total messages timed out in Channel Protocol Module;
n -- Total data packet retransmissions;
p -- Total data packet first transmissions;

q -- Total acceptable data packets heard for this site, but discarded because of buffer shortage or host output queue full;
r -- Total acceptable data packets heard for this site and accepted;
s -- Total acceptable data packets heard for others.

HOST TRAFFIC REPORT FORMAT

s time PS = a/b PR = c/d HI = e/f MS = g/h/i MR = j/k

where:

s -- Reporting site identifier (E=Etam, G=Goonhilly, T=Tanum, C=Clarksburg);
time -- Local TENEX time;

a -- Total data packet retransmissions;
b -- Total data packet first transmissions;

c -- Total packets received with hardware errors;
d -- Total good data packets received;

e -- Total Hello packets received;
f -- Total I-heard-you packets received;

g -- Total messages discarded from host output queue because holding time exceeded;
h -- Total messages sent but refused by host;
i -- Total messages sent;

j -- Total received messages discarded;
k -- Total received messages accepted.

TIMING REPORT FORMAT

s time RTT=a/b GT=c/d

where:

s -- Reporting site identifier (E=Etam, G=Goonhilly, T=Tanum, C=Clarksburg);
time -- Local TENEX time;

a -- Last Round-Trip-Time difference from previous Global-Time update
b -- Max Round-Trip-Time difference from previous Global-Time update

c -- Last Global-Time difference from previous Global-Time update
d -- Max Global-Time difference from previous Global-Time update

2.2 Hardware Modifications

As a cost-cutting move, the 48 Kb/s circuit between the Tanum Satellite IMP and the NDRE gateway was removed from service and replaced by the 9.6 Kb/s circuit formerly between the Satellite IMP and the NORSAR TIP. Consequently, no longer can the ARPANET direct connection circuit via SATNET provide connectivity between the London TIP and the NORSAR TIP; during outages of the Etam Satellite IMP, the London TIP will become isolated.

This changeover was not without problems. Originally the Norwegian PTT replaced a faulty Codex modem at NDRE. Subsequently, even though all possible modem loopbacks on the line seemed to work perfectly, data were unable to be exchanged between the Honeywell 316 at Tanum and the PDP-11/40 at NDRE. Eventually the circuit was restored to service when Norwegian personnel bypassed two data inverters on the PDP-11/40 VDH-11 drivers. Apparently the specially designed converter box used with the Muirhead 48 Kb/s modems performed this inversion in the previous configuration.

We also coordinated with COMSAT personnel to replace the the PSP terminal end-of-packet generator printed-circuit card at the Etam ground station.

3 PLURIBUS SATELLITE IMP DEVELOPMENT

The major activities during the quarter continued to be participation in the ongoing integration of the Wideband Network subsystems and enhancements to the throughput capabilities of the PSAT.

BBN had been observing a large number of checksum errors on packets sent over the satellite channel during the early portion of May. Since measurements by Lincoln Laboratory personnel made us confident that the channel itself was capable of better performance. we began to look for problems with the other station equipment. The problem was actually solved during the Wideband Meeting (14-15 May) when BBN and Linkabit identified and corrected a bug in the Lincoln ESI which was causing bursts to be terminated prematurely. With this fix, it was possible for BBN to begin to run the Lincoln site on the satellite channel for extended periods of time with reasonable levels of performance. Intervals from several hours to a couple of days without any subsystem failure operating at 772 kilosymbols BPSK were common. In most cases. the recovery software in the PSAT would reset the ESI and bring the site back on the channel after a crash.

During the second half of May we also looped HDX speech over the satellite channel for the first time. The speech parcels. originated and received by Lincoln voice terminals on different

Lexnets attached to a miniconcentrator gateway, were transmitted as PSAT stream messages in a stream manually set up for Lincoln by the PSAT. Toward the end of the month Linkabit went to ISI to correct the premature burst termination problem fixed earlier in the month at Lincoln.

Our goal in June was to bring up the ISI site on its own as we had done at Lincoln. convince ourselves that it was working reliably on the channel, and then try to integrate it with the Lincoln site to checkout multisite operation. At the beginning of the month we noticed a problem working with the recently modified ESI. We had planned to try to correct this problem jointly with Linkabit when the PSAT at ISI began to experience hardware problems. Unfortunately, this problem proved to be particularly elusive and it took nearly the last three weeks of June to track down the difficulty to a bad cable in one of the bus couplers. We did continue to operate the Lincoln site on the channel as much as possible throughout the month, however, in order to investigate the robustness of the site subsystem and the types of problems that were causing the site to occasionally reinitialize.

On July 1. the BBN Research Computer Center switched to a new internet kernel on BBNE. Several bugs showed up as a result of this switchover which affected interactions between BBNE and

the PSATs. Errors in the PSAT loader tapes and in the U program prevented remote reloading of the PSATs over the ARPANET. A third bug in the RECORDER program prevented us from monitoring the PSATs correctly. All of these problems were cleaned up by about mid-July when the PSAT at ISI was again ready to be checked out with the ESI. Since Linkabit was planning to bring up the first ADM at ISI within a couple of weeks, however, they were not interested in trying to debug the interface between the PSAT and the old hardware. Work at ISI was. therefore. delayed until the last week in July when Steve Groff went out to ISI and worked with Linkabit people on ADM/PSAT integration. This integration appeared to go well although the site was not brought up on the satellite channel during the week. All of the loopback modes except TRLOOP, CIPLOOP and loopback via the Earth Terminal were verified. The inability to get up on the satellite channel was apparently due to some problem in either the ESI or Earth Terminal which Linkabit indicated that they would pursue with Western Union. A noise problem on the PSAT/ESI interface control lines showed up for the first time at ISI in July. It appears to be due to inadequate termination circuitry which BBN and Linkabit need to work out together. A "hardware patch" installed during the ADM integration permits us to move forward in the interim.

The DCEC site which was previously inaccessible for a combination of reasons became a more usable site in the Wideband

Network during the second half of July. A bug in the UNIX-based EDN gateway which discarded packets and thus prevented us from loading the PSAT was fixed in the middle of the month. (Note: DCEC has now switched to a standard internet gateway which also appears to work correctly.) The DCEC PSAT was brought up on the satellite channel for the first time on July 13 in conjunction with a trip by BBN personnel to DCEC. For one week (July 6-10) no satellite channel testing was possible due to the 96-hour "48-hour test".

As indicated above, the other primary activity during the quarter was work directed at understanding and improving the performance of the PSAT. A more streamlined version of the Channel Protocol Module software was developed and installed in the three PSATs in the field. Based on experience gained during the integration activities at Lincoln Laboratory, we also developed a more efficient version of the Host Protocol Module (HPM) software. These HPM modifications will be debugged and installed in the field in the near future. There are some fundamental packet-per-second throughput limits implied by the nature of the HPM message processing. These limits are discussed in section 3.1 below.

We made significant progress with the development of the SuperSUE poller during the quarter. Toward the end of May, the

first set of PROMs were burned in preparation for initial debugging of this board. In June we began initial debugging of the poller card on the PSAT processor bus. The hardware was then modified to operate on the PSAT I/O bus. The CPM software was augmented to support the new formats required for operation with the poller. During checkout of the poller on the I/O bus, a problem in the basic clock circuitry on the card was identified and corrected. During July, most of the bugs in the basic microcode were eliminated and the device began to work in conjunction with the standard PSAT software. Enhancement of the basic software to (1) poll two devices rather than one and (2) poke the Pluribus PID on a per-message rather than a per-packet basis remain to be implemented. In addition, we plan to carry out some additional testing of the unit before additional pollers are constructed and deployed in the field. In a related development, we completed the construction of the additional SMI cards that were to be manufactured this year and installed one of them (providing redundant capability) at ISI.

A review of the problems with the current PSAT/ESI interface and an analysis of the fundamental alternatives to the current approach were begun in June. This work should eventually lead to the detailed specification of a second-generation interface design. Some preliminary thoughts in this area are contained in section 3.2 below.

On 14-15 May BBN participated in the Wideband Meeting at Lincoln Laboratory. As an outgrowth of the discussions during this meeting BBN, Lincoln, and Linkabit personnel met at BBN on 15 June to have more detailed discussions on a range of subjects related to monitoring and control of the Wideband Network.

3.1 PSAT Message Processing Limitations

Up until recently, PSAT throughput discussions have focused on the capability of the PSAT to support traffic flows quoted in bits/second. We believe that the PSAT will support a throughput (total host bits in and out) of about 1.5 Mbps for maximal size messages. An equally important measure of throughput, however, is the PSAT capacity in packets per second.

One can develop some insight into the fundamental message/packet processing limits of the PSAT by analyzing the computations necessary to handle a single message by various modules of the software. Consider first the case of messages arriving at the PSAT from a single host. As an upper bound on the achievable host throughput, we note that the most computationally demanding uplink task, HOSTIN, requires about 650 instructions to be executed for every arriving message. Since the effective instruction execution time in the PSAT is about 7 microseconds. HOSTIN can support a maximum of about 220

packets/second if there is a processor continuously available to run it. Multiple host subscribers can simultaneously execute parallel HOSTIN processes so that higher throughput is achievable. This requires that there are additional SUE processors available to run the parallel processes. In the current environment, however, there is only a single host at each PSAT site so that the 220 originated packets/second upper bound still stands. We are looking into the possibility of changing the control structure to permit multiple incarnations of HOSTIN to operate on different messages from the same host. but this too only provides an advantage in the case of available processing hardware.

A similar analysis of the execution time for the SATIN strip, the most computationally demanding downlink task. leads one to a downlink processing limit of 250 packets/second. Adding the uplink and downlink constraints to be compatible with our existing definition of throughput. we get a "theoretical" limit of 470 packets/second per PSAT assuming a single host subscriber.

3.1.1 PSAT/ESI Functionality and Interface Issues

In the current Wideband Packet Satellite Network design. channel related processing functions are split between the Pluribus Satellite IMP (PSAT) supplied by BBN and the Earth

Station Interface (ESI) supplied by Linkabit Corporation. The ESI implements the burst modem, codec, and modem/codec control functions while the PSAT synchronizes network timing, performs channel scheduling, interfaces to host subscribers and coordinates network management and control. The existing interface between the PSAT and the ESI is defined in W-Note 13 and the PSAT Technical Report (BBN Report No. 4469). For some time now, there has been a growing belief shared by both BBN and Linkabit that this interface specification should be improved as part of any next-generation system development.

A basic problem with the current design is associated with the requirement that the ESI be able to recognize and decode bursts on the downlink without any *a priori* information. In general, the burst length contained in each control packet allows the ESI to re-arm the downlink modem at the end of each burst. In addition, the control packet statewords tell the ESI what control signals to pass to the codec for each segment of the burst. For small bursts, however, the burst length word may not have been decoded by the time that the burst completes and the next burst arrives. Similarly, under certain conditions the latency for determining what coding command to send to the codec may be so large that data within the burst is lost. Linkabit is currently evaluating the use of a unique trailing word in conjunction with measurements of carrier energy to address the

burst recognition problem. The stateword decoding latency problem remains to be addressed.

One general set of solutions to the downlink processing problem involves getting information to the downlink ESI describing exactly what it should expect prior to burst reception. Either the ESI can do the computation necessary to figure out this information itself or the PSAT can convey this information to the ESI. In either case, a revision of the current functional split between the PSAT and the ESI may be necessary. In the remainder of this section, we enumerate a set of approaches and summarize some preliminary thoughts on the possible PSAT/ESI interface alternatives.

1. Status Quo - Simply live with what we have for the next generation design. The ESI would have to try harder to handle its task as currently defined. This might require the addition of buffering, processing capacity, or both.

2. Incremental Modification of Status Quo - Although the details of this solution require some work, the basic idea is to keep the existing functional split and work to modify the interface specification in W-Note 13. This may clean up the interface but would again not address the downlink processing problem.

3. Status Quo with Limitations on Burst Structure - The idea here is to simplify the difficulty of the downlink processing by restricting the modulation type and coding rate to be uniform within a burst. One can combine this alternative with #2 above to clean up the interface specification as well. This alternative is likely to be viewed as too restrictive, however. Moreover, it does not solve the modem's burst recognition problem.

4. CPM in the ESI - Putting the Channel Protocol Module (CPM) in the ESI has been an approach suggested by Linkabit. This approach could allow the ESI to have all the required channel schedule information but at the expense of making the PSAT essentially disappear. This approach seems rather extreme given the magnitude of the CPM software development required. In addition, splitting the CPM and HPM modules will, we believe, lead to serious problems in the future with regard to evolving capabilities for flow control, congestion control, and network management.

5. ICCU in the PSAT - This approach is the BBN-biased analog of the previous one. Under this approach, the PSAT would be expanded to assume some or all of the functions of the current ESI ICCU. The primary new functions that the PSAT would perform are arming the burst modem and control of the

codec.

6. ESI Manages Global Timekeeping - Under this approach, the ESI functionality would be expanded to include management of networkwide global time. The major advantage of this approach, in addition to opening the possibility for a cleaner less time critical PSAT/ESI interface and off-loading the PSAT, is the potential for the PSAT informing the ESI about what bursts to expect on the downlink. Whether the ESI would establish global time using the same algorithm currently used by the PSATs or some other technique remains to be determined. In either case, the new function that the ESI must take on is nontrivial. This approach is likely to have subtle problems lurking since we are removing such a critical element of the PSAT application.

7. ESI Manages Local Time - Under this approach, the ESI would do all of the precise time measurement but the PSAT would retain the global timekeeping function. The PSAT would still be able to inform the ESI of the channel schedule and burst structure in terms of local time. This approach would also simplify the PSAT/ESI interface by moving the time-related portion of the SMI function into the ESI where it must be resident anyway. The PSAT and ESI can exchange all data over a single serial port. T&M and control data are exchanged

over a second serial port.

Alternatives 4, 5, 6, and 7 all require the downlink ESI to obtain complete information on the burst structure prior to receipt of the burst to solve the downlink processing problem. For datagrams, the obvious place to transmit this information is in the associated reservation by expanding the format which currently specifies only total burst length. Unfortunately, if we let our new reservation format be two words per packet to define coding and modulation type (we simply move the stateword which used to be on the packet to the reservation). the reservations become variable length due to variable number of packets per burst.

A solution to this problem which has some additional beneficial side effects is for the PSATs to rearrange the burst structure so that all burst segments with the same coding and modulation type are adjacent. The maximum number of states/burst in the current environment would, therefore, be 8 (4 coding rates * 2 modulations types). With a fixed size reservation of 8 words, one could specify the structure of an entire burst. This represents a significant increase in overhead for single packet bursts but is about as efficient as the current design for bursts with an average of 3 or 4 packets. At the expense of some additional restrictions, one can make the overheads with and

without rearrangement even closer. For example, if we were to require that only a single modulation type be applied to each burst, the maximum number of states/burst would be 4 and the additional overhead for the proposed scheme would be 1 word more per reservation/datagram in the worst case (single packet bursts). While the PSAT is currently unable to do the processing required to rearrange bursts, a Butterfly multiprocessor-based PSAT probably does have the necessary processing power. In addition, burst rearrangement may actually simplify the computational load associated with channel scheduling for large multipacket bursts. This point needs to be analyzed more closely in the context of the I/O architecture of the Butterfly multiprocessor.

Sending the fine structure of a burst over the channel must be handled differently in the case of streams than it is in the case of datagrams. For streams, no per-burst reservation is transmitted and there can be arbitrary variations in the burst structure within a given channel stream. One method which appears at least plausible is for the PSAT originating a stream burst to delay that burst for one frame and send the structure of that burst in the control packet of the burst in the previous frame.

The use of *a priori* information for burst reception introduces some additional problems for initial acquisition and PSAT synchronization. There must be some additional mechanism provided to get the system started so that a station can receive the initial channel schedule and burst structure information. Clearly, this entire area needs additional study before the best alternative can be selected with confidence.

4 REMOTE SITE MAINTENANCE

4.1 Software Distribution Methods

The UNIX system and its major utilities tend to be quite dynamic, with corrections and enhancements constantly being produced. In general, the more or less continuous distribution of updates is awkward; on the other hand, users wish to have access to the latest, and presumably best, software. Over the past year, BBN has generally batched the installations in rather large units, and used site visits to perform these functions. Recently, the problem of scheduling routine distributions to the Remote Site Modules over the network has been examined in some detail. One would think that maintaining a consistent environment in this way would be easy; our examination of the problem has shown this not to be true.

There are two basic methods of keeping the software on several machines synchronized: transmission of a complete copy of the software at specified periods, and distribution of incremental updates containing only new and revised programs.

The complete copy method, using disk to disk, tape, or a network, is simpler. The major problem is that even for a small number of systems the time required becomes prohibitively expensive, especially when the distribution cycle is short and

the number of actual modifications small. For example, if one distributes only binary images of the controlled software. it requires about an hour per machine to copy and verify that all is well, even when the systems are connected to the same IMP. Cross-country transmission and encryption devices, not to mention satellite links. reduce the available bandwidth significantly. If one is to be certain the copy is exact then the old directories should be removed, to avoid problems of "leftovers". For example, suppose program 'prog' is moved from /bin moved to /usr/bin directory. If the programs are simply copied. then 'prog' would be added to /usr/bin but never removed from /bin.

The incremental method is conceptually cleaner, and may occur without disruption to the recipient systems. In the following discussion, it is assumed that the modifications are prepared on a single system, and that they are installed as they become ready. The update is to be sent periodically, and should not include brand-new installations on the master system, since these have not been subjected to routine operation. Extensions of these procedures will be required in the case that development takes place at more than one site.

There are three basic types of changes: addition of new files. removal or renaming files. and insertion of new links to old files. Experiments have been performed to explore these

procedures. A shell script using the 'find' command locates the new files. This script identifies files older than N days and newer than M days, where N and M are parameters. Another shell script uses the 'tar' command to bundle the files into one large file which can be shipped over the network to the other systems. In order to detect files that have been moved or renamed. and to locate new links. a file containing the long form of the directory listing (along with any subdirectories) is maintained. This file is compared with the installed version. A command file which performs the necessary operations is hand-crafted, and is sent off to the receiving systems. After the files have been transferred, one must login and run the command files to 'untar' the new files and execute modification commands. News messages must be posted and the installation noted in the message of the day.

Although this technique is usable, it requires a significant amount of operator intervention to work. The division between the file systems on the machines is one factor which increases the complexity of the task; another is the the high visibility of the network. If the file systems were logically unified. one could move file 'data' from a directory on machine A to another directory on machine B by typing 'mv /A/usr/uid1/data /B/usr/uid2/data.new'. Alternatively, one could establish an installation protocol which could move the data silently, without

additional logins. This would function like mail, for example. One important difference would be the support of an installation queue in which the requests would be held for N days. and the addition of those link/unlink commands which would allow files to be removed or renamed.

4.2 Info System

The 'info' system described in the last QTR has been extended to support an additional access mechanism which uses a hash-coded index to the basic data base, and to a bug reporting system.

The initial implementation of 'info' used a single method of access to the information elements. These are arranged in the subdirectories of /usr/info. The basic hierarchical pattern of access is modified by allowing direct transfers from one branch to another. Any scheme of this sort depends on the user developing a certain facility with the system, or being willing to spend time exploring the tree whenever he needs a new piece of information. But this would defeat the purpose of the 'info' command: it is intended to be a fast, easy way for a user to locate relevant information.

In order to meet this objective, an additional way of finding the correct node in the tree is needed. The Version 7 UNIX system has a set of programs which may be used to maintain a hash-coded index; these are described in "Some Applications of Inverted Indexes on the UNIX System", by M. E. Lesk of Bell Laboratories; this document is included in Volume 2A of the the standard Bell distribution. The new facility is based on these programs.

This automated indexing scheme breaks the files up into pieces. extracts the words which are used, and hashes them into an index. The distributed programs have several shortcomings which have been corrected.

1. Only the first 500 characters in each text block are scanned by the standard programs. Restructuring the programs allows the entire text block to be used.

2. The distributed programs do not discard repeated usage of the same term in a text block when constructing the index.

3. The distributed programs do not allow simultaneous retrieval of a text block and its identification, making it impossible to know where to find the interesting information.

4. The distributed programs support only two types of text block: paragraphs and complete files. They have been extended to support BBN-UNIX message files. and an extension to UNIX-style manual pages is planned.

Two new info commands are based on this improved automatic indexing system. These are 'hunt', which finds the information blocks which match a set of terms. and 'bug', which searches the bug-reporting system's mailboxes for any messages which contain the terms.

Experience to date with this system suggests that the following steps be taken next:

1. The 'hunt' command be extended to search the system news file. the UNIX manual, and other documents. This would then provide a uniform method for accessing the on-line documentation.

2. The 'info' command be re-coded as a C program. The current shell file implementation is quite slow, and users quickly become frustrated with it.

3. A mechanism for creating the information packets be developed. This program would take a formatted input file that contained the information about the proper place in the hierarchy and the cue information, along

with cross-tree information, and update the information tree.

4.3 Archival Storage System

4.3.1 Objectives

There is a need, within any dynamically modified system, to store copies of text in such a way that one can call them back as needed. Such an archival storage system may also be used to handle user data which may in the future be needed again, but which should be removed from system storage until that date. The term "archival storage system" is used here to refer to a system in which the user explicitly identifies files which he wishes preserved for an extended period of time. This is in contrast to an automatic incremental backup system, which UNIX already provides. Such a system is described below; implementation is currently underway.

The system makes provision for moving files to tapes or to other hosts. It preserves the ownership, permission, and date attributes of the archived files. It provides a command to query the database to determine what versions of a file have been archived, and it provides optional storage compression mechanisms.

From the user's point of view, a good archival storage system should have the following attributes:

- The system should provide facilities for examining and changing the state of archived files, and its policies toward such actions should be consonant with those of the operating system.

- The user should be able to find out what backup copies are available, when they were made, what their time-last-modified is, etc., and to retrieve any version. In particular, the user should be able retrieve the last backup before this current file, iterating this process as many times as desired until he obtains the file he wants.

- The user should be capable of backing up any file system entity. For UNIX, this means being able to back up regular files, directories (by backing up each of the files contained), or UNIX "special files" (by saving the attributes of the file).

- The file attributes should be restored by 'retrieve'. This includes the time-last-modified and the time-last-accessed. The "time-last-created", which actually refers to the last time the inode was changed, should be set to the current time, so that the incremental backup facility will be

triggered by the retrieval. The permissions should also be restored.

- The retrieval command should make provision for restoring a file to a different name than the one under which it was archived. Similarly, it should be possible to print the file to standard output (without actually retrieving it) for purposes of identification and comparison.

- The act of moving archived files offline should be relatively transparent to the user; the command which lists the files should list such files as well as those which are still available, with a parenthetical note, and the command which retrieves files should be capable of automatically initiating the process of retrieving these more remotely archived files.

- For user convenience, provision should be made for specifying, as a file is being archived, that it may be moved offline as soon as practicable.

- For operator convenience, the commands which manage the movement of files to and from offline storage should be simple to use.

- For administrator convenience, a wide variety of policies with respect to the movement of files offline should be

permitted.

4.3.2 Description

It will appear that the 'backup' command simply saves a copy of the specified file(s), without altering either the user's own files or any other files which may have been backed up under the same pathname. 'Retrieve' restores a named file to the spot it came from, unless otherwise specified, and overwrites whatever is there.

The database used for the archiving contains a record of each backed-up file and its status. When a file is backed up, a copy is placed in the backup filesystem in a directory whose name is that of the file being archived. Files within the backup hierarchy may be moved offline, or to another host. Neither of these actions affects the original copy of the file. The user's regular filesystem has no knowledge of archived files; instead, additional commands are available to interrogate the backup system about the status of any particular file, or to modify that status.

A user may backup as many copies of a particular file as he wishes. Each time, a new entry in the backup hierarchy is made, using an internal name which the user need never know. Users

select among the versions of a file by specifying the relative date of the backup. This information is given with the "ls_backup" command, similar in function to "ls -l".

UNIX policies concerning ownership and protection of files are carried over to the backup system. Storage, retrieval, and similar manipulations shall be permitted if their counterparts (such as "cp") would be permitted on the original filesystem.

Directory handling differs slightly from the handling of other files. 'Backup' archives a directory (if the "recursive" option is set) simply by saving each of the files in its hierarchy; an empty directory cannot be backed up. 'Retrieve' does not restore directories to their exact pre-backup state: all files in the backed-up hierarchy will be restored, but files in the target hierarchy which were added after the backup will not be removed. Also, no attempt is made to rebuild links which may have existed at the time of the backup.

This system is, for the most part, ignorant of links. It does warn the user if the number of links to a file has changed between the time it was backed-up and the time a retrieval was requested. If the link count of the target file at the time of retrieval is greater than one, the user will be asked whether the links should be broken before the file is overwritten with the backup copy.

After a file has been archived, it may be moved offline onto tape, or onto another host's backup filesystem. The status information remains regardless of the movement of the file, and record is kept of the location of the backup file. Files which have been moved offline successfully, either by user or administrative request, are deleted from the backup hierarchy; they are still controlled and listed by the backup/retrieve family of commands. The original copy in the regular filesystem remains unaffected, of course.

4.3.3 Command Summary

User Commands:

backup file ... [-d] [-r] [-!r] [-diff] [-off]
(Make backup copies of the indicated files)

ls_backup file ... [selector] [-nm] [-l] [-a]
(Find out about the indicated backup copies)

retrieve file ... [selector] [-as <name>] [-in dir] [-o]
[-bl] [-!bl] [-off] [-!off] [-show]
(Retrieve the indicated files)

rm_backup file ... [selector]
(Expunge the indicated files from the backup hierarchy)

move backup file ... [selector] (-on ¦ -off)

(Change the online/offline status of backed-up files)

Operator Commands:

moved_offline tapeno [-t storage_type]

(Used after offline copies of files marked for offline storage have been made.)

move online [directory]

(Print the names of files to be brought online; to be used in conjunction with tar commands.)

moved_online [directory]

(Used after tape copies of archived files have been retrieved into the staging area; moves them into their places in the archive hierarchy and performs pending retrieve requests to give them back to users.)

5 INTERNET MAINTENANCE AND DEVELOPMENT

Internet research and development efforts during the quarter were in three areas:

- o operational support
- o design and planning for re-implementation of the gateways
- o enhancements to the current system.

These efforts were carried out in the context of a long-term plan which was developed to effect the transition of the internet from a research vehicle to an operational network. This plan primarily involves the re-implementation of gateways as an integrated system, using new hardware and software. This will permit gateways to be configured which have greatly increased buffering capacity, and which include additional mechanisms which are critical to behavior as an operational system, in particular for monitoring and control.

The resultant gateway system is intended to be installed as replacements for the current PDP-11 gateways. Since this cannot be done simultaneously at all sites, a transition plan to provide for orderly changes in the system is necessary. In addition, the ability of the system to expand to a potentially large number of gateways, and the ability to interoperate with separate gateway systems, must be addressed.

The transition plan which we have developed involves several stages:

1. final enhancements to the current gateways
2. design, implementation, and deployment of replacement gateways
3. introduction of subsequent system software releases which improve performance, or increase functionality.

The first stage involves those tasks which are necessary in support of current user activity, and activities which occur before the second stage gateways have been deployed. Some examples are the provision of a SATNET loader, support of a V2LNI local network interface, and conversion to the ICMP protocol specification. We anticipate that this stage will be largely complete by early in CY82.

The second stage will involve the implementation of new gateway hardware and software. Our plan is to use C/70 hardware, with new software support drawing as much as possible on the efforts of other projects. A major goal of this implementation effort will be the inclusion of features for maintainability and support, such as are found in the ARPANET IMPs. More important is the introduction of mechanisms which will permit the interoperability of several different gateway-gateway interaction mechanisms, while still presenting a uniform interface to the user.

This latter facility will be important for two reasons. First, it will permit the new gateways to be introduced gradually, since they will be able to interoperate with the current gateways. Second, it will permit research to continue to explore new ideas in mechanisms and protocols for the Internet, by establishing a mechanism which permits various systems to effectively coexist. This will permit new ideas to be tested, and will also permit research to be performed simultaneously by various groups if desired.

We anticipate that the gateway release which occurs as part of this second stage will include only some of the new mechanisms which we consider promising. The primary goal of the second stage will be to quickly release new gateway systems which provide increased performance by adequate buffering, and increased maintainability by mechanisms for monitoring, testing, and control.

The third stage of the transition plan will involve a series of experiments, software releases, and possibly development of alternative hardware implementations. This stage will serve as a testing ground for the ideas developed in the research community, including the research being carried out by the ARPANET Routing Study which is investigating the Internet modelled as a network itself.

5.1 Operational Support

Activities in support of the current gateway system included a mix of diagnostic and repair functions. Two notable problems were addressed, concerning the UCL and Comsat gateways. In addition, further work was done on the VAN gateway.

The UCL problem was initially reported as a loss of connectivity between programs running on ISIE and networks 11, 25, and 35. This problem was observed to happen at various times, and appear and disappear sporadically. The problem was finally traced to an event which would consistently create the loss of connectivity. Whenever the BBN gateway (between ARPANET and SATNET) was reinitialized, connectivity to those three nets was lost, even though it actually existed.

The root of the problem was finally traced to a slight bug in the TOPS-20 TCP, which was incorrectly processing GGP echo reply packets which are used to determine the status of gateways.

This problem was in fact not in the gateways per se, but rather in code peripheral to the gateway system. The experience emphasized the need for sophisticated monitoring, testing, and maintenance facilities to perform fault isolation. In cases such as this one, where the problem appears to be intermittent, it is especially necessary to have built-in monitoring and fault

isolation tools.

The Comsat gateway developed a similar intermittent problem during the quarter, which is manifested by garbled packets. This has progressed to a point where the gateway cannot now be loaded. Investigation of this problem continues.

The VAN gateway was completed during the quarter, to a point where further testing requires access to a port on Telenet. The gateway successfully passed level 3 (packet level) tests in May. We have had several exchanges by network mail with the European experimenters who are implementing a similar device to serve as an exit counterpart to the VAN gateway, exploring plans for testing and operational use.

5.2 Design Work

Work on the design of the Internet as a system was done in conjunction with the efforts of the ARPANET Routing Study, Internet task. Basically this work has involved looking at the history of the current Internet, and analyzing the day-to-day activities which occur as part of the operational support. We have used these inputs to assist in developing a model of the Internet as itself a network, where the gateways are packet switches, and they are connected by "lines" which happen to be

other packet-switched networks.

Work during this quarter continued the efforts to apply traditional network design methodology, and the experience of the ARPANET, SATNET, and other networks, in the analysis of the Internet as a network. We have attempted to identify the model of a network as applied to the Internet, facets of the network model which seem to apply directly to the Internet, and facets of that model which do not directly apply, primarily because of the differences between the behavior of a "line" which is a network, and the behavior of more traditional "lines" such as telephone links or satellite channels.

We expect that future work will continue to refine this model, and will investigate several specific designs of the mechanisms and protocols which must control gateway interactions.

Work during the quarter was documented in a series of Internet Experimental Notes, numbers 182-4 and 187-9.

X.3 Enhancements to Current Gateways

Two major implementation tasks were pursued during the quarter, in addition to plans which were developed for the other enhancements which will be needed to the current gateways.

The SATNET loader was completed during the quarter, and

tested using the test Satellite IMP at BBN (SIMPA). This loader is necessary to permit gateways to be re-initialized via the SATNET path. Currently, loading is done via the ARPANET path. This technique will no longer be available when lines 41 and 77 are removed from service in fall 1981.

The second major implementation task involved the NDRE gateway. In order to provide network service at this site after line 42 is removed, a ring-style local network has been procured by NDRE for installation in Norway. To support this network access, the gateway is being enhanced to include hardware and software for interfacing to the V2LNI ring network, manufactured by Proteon. We have ordered, and received partial delivery on, PDP-11 and LSI-11 peripherals to construct a test and development V2LNI network at BBN, and have begun planning for changes to the TIU and gateway to interface to this network.

6 MOBILE ACCESS TERMINAL NETWORK

As part of our participation in the development of the Mobile Access Terminal (MAT) and the MAT Satellite Network (MATNET) during the last quarter, we finished the preliminary system integration at E-Systems. ECI Division, in St. Petersburg, Florida, and subsequently began the system integration within the Advanced Command and Control Architectural Testbed (ACCAT) experiment at the Naval Ocean Systems Center (NOSC) in San Diego, California. Below are described some of the accompanying events.

Upon termination of the preliminary testing at ECI, two complete Red subsystems, except for the BBN C/30 packet switch processor electronic boards, were shipped to NOSC. The latter were shipped to the BBNCC for the addition of engineering retrofits prerequisite to the purchase of field maintenance. Eventually, refurbished boards from C/30 #1 were installed into the BBN backroom MATNET testbed chassis, while refurbished boards from C/30 #2 and #3 were installed into the two C/30 chassis at NOSC. (C/30 #1 has sockets for chip insertion, while #2 and #3 have soldered-on chips and therefore are considered more durable.)

In the interim between the preliminary testing at ECI and the system testing at NOSC, we designed and built a rudimentary

digital satellite channel simulator for interfacing C/30 MATNET Satellite IMPs (Red processors) in the absence of cryptos, Black processors, and AN/WSC-3 radios. The device generates a data clock and logically OR's satellite output data from two C/30s for generating the C/30 satellite input data. With this device, a two-site network using effectively a ground level satellite can be created for testing MATNET operations, including the Terminal Interface Unit (TIU), the gateway, the MATNET Monitoring and Control Center (MMCC), and the Command Center Network (CCN) interfaces.

Also in the interim, new Satellite IMP, TIU, and gateway software incorporating patches discovered necessary during the testing at ECI was assembled and written on tape cassettes for loading C/30 machines and LSI-11/03 machines. The new software also incorporated new C/30 system microcode.

Despite an inordinate difficulty in getting NOSC security people to recognize our security clearances to the extent that copies had to be facsimilied from NAVELEX to NOSC at the last minute, we were able to install the two Red subsystems at NOSC as scheduled. In the process of installing the Red subsystems, we discovered a casualty in the hardware shipment from ECI to NOSC; namely, a Lambda 12 volt power supply for a C/30 failed to put out any voltage. Continued operation necessitated our borrowing

a laboratory supply from NOSC, until the BBNCC could make a replacement unit available a week later.

Of the two cables between the Red subsystems and the cryptos, one failed to pass GOSIG correctly. Furthermore, only one pair of cryptos sufficient for interfacing only one MAT was available for MATNET testing. Therefore, network operation with more than one site required the digital satellite channel simulator, bypassing cryptos, Black processors. and AN/WSC-3 radios. All the inadequacies described above have subsequently been remedied (extra cryptos dedicated to support the second MAT were shipped from ECI to NOSC after we had left the site).

While at NOSC, we installed RECORDER, MONITOR, QUERY, and EXPAK in the TOPS-20 to serve as the MMCC and modified the host address tables in the TIU and the gateway to reflect their new environment. After the installation was completed, we routinely were able to provide to the shipboard MAT user MONITOR reports, the processing and routing of which involve data

(1) generated by the MATs,
(2) sent via the gateway and the PLI to the TOPS-20,
(3) processed by the MMCC programs in the TOPS-20,
(4) presented to the Telnet/TCP process in the TOPS-20,
(5) sent via the PLI and the gateway to the shore-based MAT,
(6) sent via the digital satellite channel simulator to the

shipboard MAT,

(7) presented to the Telnet/TCP process in the shipboard TIU.

The above sequence of steps exercises all the fundamental network functions invoked by MATNET and as such forms a comprehensive test of network operation.

When we had left the site, we had successfully tested one MAT station with the FLEETSAT satellite on two separate occasions. (Lack of cryptos prevented testing both MATs through the satellite.) The MMCC data collection and processing programs in the TOPS-20 provided a record for both demonstrating the success of the tests and evaluating test performance.

6.1 Contention Testing

While at ECI, we conducted contention testing, the purpose of which is to determine what is received when control packets are transmitted by both stations simultaneously. Such a situation occurs normally with the use of CPODA (Contention Priority-Oriented Demand-Assigned) but not FPODA (Fixed Priority-Oriented Demand-Assigned) channel protocol. Since MATNET implements distributive channel control, satisfactory network operation requires that all stations either receive or

fail to hear the same channel request packet. Unsatisfactory network operation results when a station fails to receive a channel request packet which other stations hear.

The test configuration incorporated two complete MAT stations with AN/WSC-3 radios interfaced through the ECI-built RF satellite channel simulator. By insertion of a patch into the Satellite IMP software, we forced each station to send a control packet in the same Virtual Slot (10.24 millisecond interval) every frame, resulting in packet contention every frame. Statistics collected by the Black processor on single-block packets form the basis of test results, since a control packet is sent as a single-block packet. The test, which was repeated at various transmit power levels with consistent results, indicates that differences in AN/WSC-3 radio performance caused one station to receive several percent more packets than the other. If distributive channel control with CPODA channel protocol were running, then the sites would be out of reservation synchronization most of the time; therefore, our plans are to run future tests of MATNET with FPODA channel protocol only.

Because in the FPODA frame structure a separate control slot is assigned to each site for its dedicated use in requesting channel allocation, the control subframe must grow as the network increases in size. Currently the frame allocation incorporates 3

control slots to accommodate 3 sites. where each control slot is 40 milliseconds long. Once 6 sites are fielded. the control subframe must be expanded to 6 control slots for a total of 240 milliseconds. Consequently, a PODA frame length of 310 milliseconds. the current SATNET default, is too small, resulting in control overhead of 240/310 = 77%. Thus, we have increased the PODA frame length to 610 milliseconds. which is about twice the SATNET default and about two satellite round trip times. Framing delays now dominate satellite channel delays.

6.2 TIU Host Table Entries

For user convenience, we inserted entries into the MATNET TIU host table to allow the user to refer to specific MATNET, ACCAT, and CCN host addresses symbolically. These host table names along with their numerical equivalents are summarized here.

In the table below, the first three columns provide the octal, the decimal, and the Host/IMP equivalent specifications of the host address, respectively. The fourth column presents the symbolic host name(s). The last two columns provide the decimal value and the symbolic name of the network on which the host is a member.

HOST				NETWORK	
OCT	DEC	Host/IMP	NAME	DEC	NAME
371	249	3/57	ACCAT-TENEX, TENEX	10	ARPANET
372	250	3/58	ACCAT-TOPS20, TOPS20	10	ARPANET
373	251	3/59	ACCAT-UNIX, UNIX	10	ARPANET
374	252	3/60	ACCAT4	10	ARPANET
375	253	3/61	GUARD	10	ARPANET
376	254	3/62	ACCAT6	10	ARPANET
377	255	3/63	ACCAT7	10	ARPANET
271	185	2/57	CINCPACFLT, CPF	10	ARPANET
311	201	3/9	BBN-RSM	10	ARPANET
321	209	3/17	FNOC	10	ARPANET
331	217	3/25	NPS-UNIX	10	ARPANET
341	225	3/33	MOFFETT-ARC, ARC	10	ARPANET
351	233	3/41	SPEL	10	ARPANET
17	15	0/15	NIU15	27	CCN
20	16	0/16	NIU16, CCN-NDS, NDS	27	CCN
21	17	0/17	NIU17	27	CCN
22	18	0/18	NIU18	27	CCN
23	19	0/19	NIU19	27	CCN
24	20	0/20	NIU20	27	CCN
25	21	0/21	NIU21	27	CCN
26	22	0/22	NIU22	27	CCN
27	23	0/23	NIU23	27	CCN
130	88	1/24	NIU24, CCN-NSM, NSM	27	CCN
31	25	0/25	NIU25	27	CCN
32	26	0/26	NIU26	27	CCN
11	9	0/9	SHORE1-TIU	34	MATNET
31	25	0/25	SHIP2-TIU	34	MATNET
51	41	0/41	SHIP3-TIU	34	MATNET
71	57	0/57	SHIP4-TIU	34	MATNET
111	73	0/73	SHIP5-TIU	34	MATNET
131	89	0/89	SHIP6-TIU	34	MATNET

TIU Host Table

To issue a Telnet connection request to any host, the user must type on his terminal a character string having the format shown below.

@o Host (Host/IMP), Network, TCP-PORT

Items in the request are specified either decimal or symbolic; however, mixing decimal and symbolic items is acceptable. For symbolic host name inputs, the network need not be explicitly specified; the TIU host table will identify the correct network. For numeric host address inputs, ARPANET is assumed whenever the network is not specified. The Telnet port (port 23 decimal) is assumed whenever the TCP-PORT is not specified. (Note, a single decimal host address number cannot be used whenever an IMP number greater than 63 is indicated in the equivalent Host/IMP form; the Host/IMP form or the symbolic name must be used.) Some examples of connection requests are shown below, where those requests grouped together are equivalent.

```
@o Ship2,MATNET,1
@o Ship2,,1
@o 25,34,1

@o TOPS20
@o TOPS20,ARPANET,Telnet
@o 250,10,23

@o NSM
@o NSM.CCN
@o NSM.CCN,Telnet
@o 1/24,CCN
```

6.3 CCN/MAT Integration

From the very beginning, internetwork compatibility has been designed into both the CCN and MATNET through use of the ARPA sponsored internet datagram approach. Fundamental to both networks is the usage of standard networking protocols. including Internet Protocol (IP) as the transport layer, Transfer Control Protocol (TCP) as the host-to-host layer, and Telnet as the user layer. Conversions between local network protocols for each system are provided by general purpose gateways; the particular local network link protocols include the LSI-11 Ungermann Bass Parallel Interface (LUBPI) for the CCN and the ARPANET standard 1822 Host-to-IMP interface for the MATNET and the PLI. Gateways to a MAT are physically connected as hosts on the Satellite IMP.

CCN/MAT integration has three distinct stages; in all three stages, address tables within the gateways must be modified to reflect their environment. The accompanying figures specify the system configuration in each of the stages. where gateways are depicted by a rectangular box encompassing the letter G.

In the first stage (see Figure 1). gateways to the CCN and to the shore-based MAT are separately connected as hosts to the ACCAT Private Line Interface (PLI). In this configuration, the CCN is a simulated shore-based system, and the PLI serves as an intermediary for internetwork communications. All CCN traffic

destined to the shipboard MAT user must pass through MATNET, while the remaining CCN traffic is handled locally by the PLI.

In the second stage (see Figure 2). the CCN gateway is removed from the PLI and connected to the shipboard MAT for simulating a shipboard CCN system. All CCN traffic destined to the shipboard MAT user is handled locally by the shipboard MAT, while the remaining CCN traffic must pass through MATNET.

In the third stage (see Figure 3). the CCN gateway is reattached to the PLI similar to stage one for simulating a shore-based CCN system; however, an additional gateway is used to interface directly between the shore-based CCN and the shore-based MAT. All CCN traffic destined to the shipboard MAT user must pass through MATNET, while the remaining CCN traffic is handled locally by the PLI. This configuration requires changes in the CCN software to recognize that traffic destined to the MAT user should be directed to a gateway different from that used by the remaining traffic.

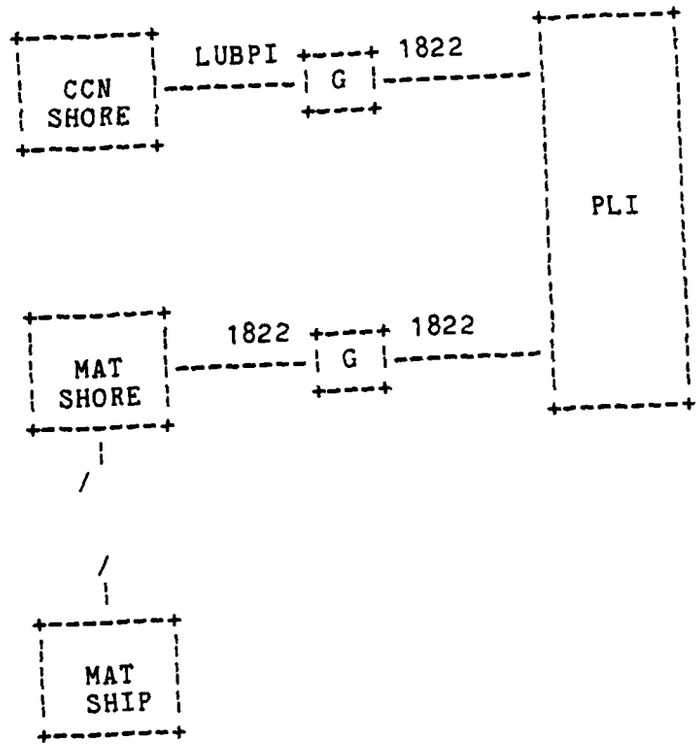

Figure 1. Stage 1 CCN/MAT Configuration

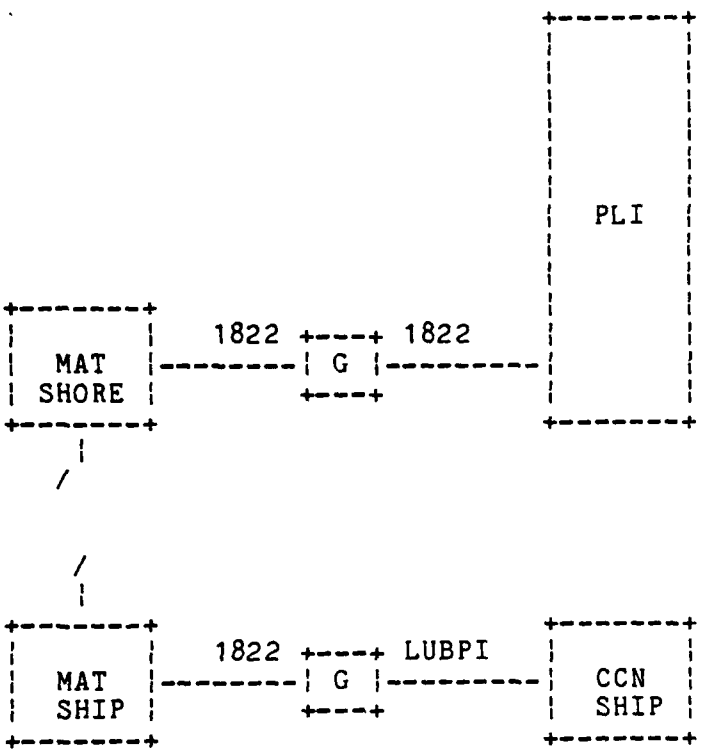

Figure 2. Stage 2 CCN/MAT Configuration

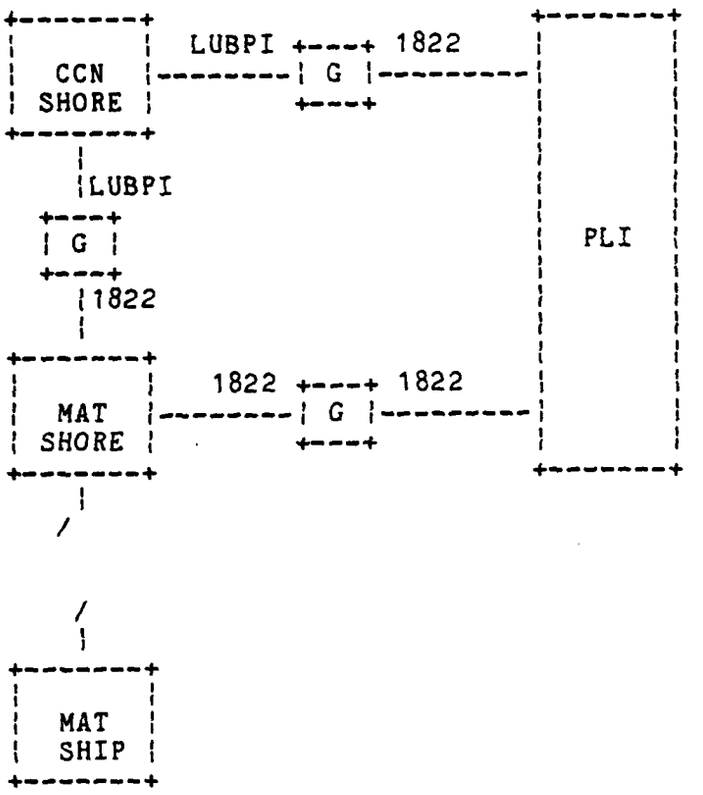

Figure 3. Stage 3 CCN/MAT Configuration

7 TCP FOR THE HP3000

The HP3000 network protocol software was completed during the last quarter. We now have software to implement the IP, TCP, TELNET, and FTP protocols. Extensive testing of the sofware modules has revealed only a few minor bugs which were easily fixed. Among the tests was a 16-hour TELNET echo test which revealed no problems.

The only significant problem encountered was a bug in the HP3000 operating system, discovered during a TCP throughput test which revealed a high data error rate. The problem was isolated to the CS3000 driver software which interfaces the TCP protocol software to the INP network interface. While the CS3000 software is designed to allow multiple buffering of network data, the operating system allows only one network input buffer to be queued at one time. An attempt to queue more buffers would result in the loss of data in all but the last buffer queued.

The buffer queueing bug made any TCP throughput tests unfeasible. Any attempt to test the actual throughput of the protocol software requires that the operating system allow multiple buffering of incoming network data. Without multiple buffering the incoming data rate is severly reduced to the speed with which the operating system can pass a buffer from the INP to the protocol software. This speed is far slower than the

throughput rate of the protocol software.

Fortunately, HP has also discovered the bug and fixed it, and we have just received a new version of the MPE operating system which incorporates the fix. This will allow us to begin throughput tests.

7.1 The HP3000 Security Enhancement

System security is always a prime consideration when connecting a computer to a network such as the ARPANET. The network connection greatly increases the risk of unauthorized entry to a system. Network access makes it easier to circumvent any physical security precautions which have been taken to prevent tampering with the system. Providing locks on computer rooms and even terminal rooms becomes meaningless if the system can be accessed through the phone system, as is true of the ARPANET. While a considerable effort is made to prevent unauthorized use of the ARPANET, there is no guarantee that the ARPANET is secure.

The burden of preventing unauthorized access to a system on the ARPANET is therefore left to each system's login and password scheme. If the password scheme is particularly secure and knowledge of the system's passwords is kept to a minimum number

of people, the system will be secure. If, on the other hand, the password scheme is weak or the passwords are poorly protected the system will be vulnerable to unauthorized entry.

Unfortunately the HP3000 falls into the latter category. The selection of passwords is limited by the fact that the first character must be a letter and that there is no distinction between upper and lower case letters. There is also no lower limit to the length of a password. A one letter password which would be easy to discover through trial and error is perfectly acceptable to the system. In addition, there is no attempt to use any encryption scheme, all passwords are maintained in unencrypted form in a system file. Any user with access to the file therefore automatically has access to the entire system. Finally, the prompting messages used by the system tell the user what has gone wrong if a login attempt is unsuccessful. The user is told whether the username, account name, or password was the cause of the failure. This gives and unauthorized person valuable clues as to what user names and account names are valid on the HP system. This is particularly significant for HP systems which usually use standard account names for system manager and operator accounts.

In order to overcome these problems it was decided to improve the password scheme used by HP3000. In order for the new

scheme to be feasible it has to meet three requirements. First, while the scheme may appear slightly different to the user, it should not increase the difficulty of logging into the system. Second, the scheme must maintain the passwords in an encrypted form to prevent unauthorized use. Third, since it was not practical to modify the existing password scheme, the new scheme is designed to overlay the existing scheme. Finally, the scheme must only affect users logging in through the ARPANET.

The most practical way to meet all of the above requirements is a modification of the SERVER TELNET program to add a password mechanism. The modified SERVER TELNET will prompt the user for a user name and password. Use of the SERVER TELNET program assures that only network users will be affected and that no modification of the HP3000 password scheme will be necessary.

All user passwords are encrypted using a modified form of the National Bureau of Standards Data Encryption Standard (DES) algorithm. The DES algorithm is modified to make it irreversible. The encrypted form of the password is stored in a system password file. A typical file entry, shown below, also contains the user name and an encrypted form of the line a user normally types to log into the system; the normal login line contains the username, account name, and password.

sax:j74kskfz:1jleoirowowue:

account name:encrypted password:encrypted HP3000 login line:

Under the ARPANET login scheme and normal login will proceed as follows:

1. SERVER TELNET prompts the user with the line

   LOGIN:

2. The user will respond with a user name.

3. SERVER TELNET searches the password file for the user name.

4. Without telling the user whether the user name is valid, SERVER TELNET prompts the user for a password with the line.

   PASSWORD:

5. The user responds with a password.

6. SERVER TELNET encrypts the password and compares it to the password associated with the user name.

7. If either the user name or the password is incorrect SERVER TELNET responds with the following two lines, after which the user can try again:

   LOGIN INCORRECT

   LOGIN:

8. If the user name and password are correct, SERVER TELNET unencrypts the standard HP3000 login message and writes it to the

pseudo-Teletype. This message will complete the login process and allow the user access to the system.

The password file is maintained in ASCII format so that user names can be added or deleted with the standard editor. Users can modify their passwords at any time with a password program. THe HP3000 login line is encrypted after the line is added to the file with the editor.

8 TCP-TAC

The TAC project passed several important milestones in the last quarter. The TAC was successfully tested with real users. It was also run in a C/30 computer. Drafts of a TAC User's Guide and of an IEN describing the TAC monitoring protocol were written. Also, various bugs were found and fixed.

The successful testing of TAC software with real users was accomplished by running the software in the BBN-TAC (the old TEST-TIP). We have the capability of connecting 8 users (out of a maximum of 63 that the TAC can support) at one time. As result of this testing, various bugs were found and fixed. The testing and bug fixing activity is continuing.

The TAC has been tested in a C/30 with 64K words of memory. We were able to load the machine and bring up the TAC without any problems. We plan to make a few changes to the software to use the extra memory for buffers and correspondingly increase the amount of buffer memory that each port can use.

The TAC was demonstrated at the June 1981 Internet meeting. As part of the demonstration it was used to access various systems that support TCP and was used by the attendees to read their mail. It also provided a means for demonstrating the newly developed VAX TCP. A short presentation was made at the meeting

describing the flow control algorithms employed in the TAC TCP.

A draft version of the "TAC User's Guide" was written which describes how to use the TAC plus all of the TAC commands and the messages that the TAC outputs to the user. The draft version will be distributed to the first few TAC sites. When the TAC and the manual stabilize, a final version will be produced.

A draft document describing the TAC monitoring protocol was written and will be released in the near future as a IEN; it is currently being reviewed here at BBN. The document describes the protocol that will be used to monitor TACs on various packet-switching networks. We are also planning to use the same protocol to monitor the ARPANET IMPs. It is our hope that the same host monitoring protocol can be used to monitor other hosts such as gateways.

The first operational TAC in the ARPANET will be the NCC-TAC, which will be the conversion of the NCC-TIP (BBN40). We plan for this to happen in the last two weeks of September. When this machine has been up and working for about a month we will start the process of converting other TIPs to TACs.

9 TCP FOR VAX UNIX

Activity in the VAX TCP Project this quarter centered on testing and debugging the TCP/IP implementation, finishing work on the higher level protocols (TELNET, FTP, and MTP), and developing extensions to the IP and local net layers of the implementation. The TCP/IP implementation has been in general use at BBN on the UNIX Cost Center VAX. and at University of California, Berkeley (UCB) Computer Science Research Group. Integration of the TCP/IP into the latest version of Berkeley UNIX (4.1BSD) has been completed, and this version is now in production.

In preparation for a release of the software to beta test sites in addition to UCB, a second test version has been prepared and distributed to UCB and Carnegie-Mellon University. After some experience at these two sites, we are anticipating distribution to at least four other test sites early in the coming quarter. A tentative list of these additional sites includes Stanford University, Cal Tech, Purdue University, and Lincoln Laboratories.

In addition to software development work, we participated in a meeting of the ARPA Berkeley Steering Committee, held at BBN on August 3. Topics discussed at that meeting included proposed virtual memory, file system, and interprocess communication (IPC)

enhancements to Berkeley UNIX.

9.1 Higher Level Protocol Software

User and server versions of TELNET, FTP, and MTP are currently in use. This software is being distributed to the test sites along with the TCP/IP. An effort was made to package the software for easy installation at foreign sites.

As part of this effort, and in anticipation of the extended internet address format being adopted for IP by the Internet Working Group, the library of network address translation subroutines used by network applications programs was redesigned. These functions include routines to translate between ASCII strings representing host and network names and binary representations of the internet addresses. Routines in the network library, which had been in use for older NCP versions of network applications programs, were adapted for use with the new internet address formats. Also, the package was redesigned to work with multi-homed hosts (those with more than one physical network interface), as well as with those with multiple internet addresses.

9.2 TCP/IP Development

Work is progressing on extensions to the current TCP/IP implementation. These extensions include:

(1) Generalization of the local network layer to handle multiple physical network interfaces.
(2) Handling multiple local internet addresses at the IP level.
(3) Gateway forwarding at the IP level.
(4) Handling ICMP messages from the gateways.
(5) Extending the functionality of the user interfaces to the raw local network and IP layers.

Design and coding of these extensions is complete, and debugging will continue during the coming quarter.

9.2.1 Multiple Network Interfaces

The initial TCP/IP implementation accommodated a single local network protocol layer and physical interface device driver. Specifically, there were send and receive routines for the ARPANET 1822 protocol and a driver for the Associated Computer Consultants (ACC) LH/DH-11 IMP interface. The extended implementation generalizes the interface between the IP and the local network layers to accommodate multiple local network protocols and physical network interfaces.

This extension necessitated a redesign of the IP/local network interface and the introduction of a new data structure,

called the interface control block (IFCB). The IFCB holds parameters describing the the IP/local network layer interface. It contains pointers to the device input and output message queues, pointers to local network level send, receive, and raw transmission routines, the device driver send, receive, and initialization routines, and a set of device dependent and device independent flags. All routines needed for doing local network protocol processing are called indirectly through the IFCB. Since an interface may have more than one internet address associated with it (e.g., using the logical host field in the ARPANET address to multiplex message streams), there is a separate table of local host addresses whose entries consist of an internet address for the interface, and a pointer to the corresponding IFCB.

When a packet is sent to the IP output routine, a local network address to send to is determined for the destination IP address (see below). A routing routine returns a local net address (of the destination or a gateway) and a pointer to the IFCB of the interface to be used to send the packet out on. The IP output routine then sends the packet to the local network output routine pointed at in the IFCB.

From the device driver level, the device input and output routines are called from the local network protocol routines, and

passed an IFCB pointer. The IFCB contains the device number of the associated physical interface. The driver routines use this and the input and output queue pointers to reference the appropriate physical device, and store incoming and outgoing packets. In addition, initialization has been made completely asynchronous. After the device initialization routine is called. control returns to the network input process so that other interfaces may be serviced, and initialization is completed through device or timer interrupts.

Incoming messages are handled by the drivers queueing them on the IFCB message input queue. The mainline of the network input process is awakened by the driver whenever a completed message is placed on an input queue. The mainline then calls the local network protocol input routines of each entry in the IFCB table to process the message. The input routines call on higher level protocol input routines (e.g., IP input) to continue the processing.

These modifications make it possible to have more than one local network protocol layer and multiple physical network interfaces. Currently, only the ARPANET 1822 local network protocol and the ACC LH/DH-11 driver have been implemented. Work is going on at UCB to develop a driver for the Ungerman-Bass 3Mb Ethernet interface, and it is expected that other sites will

develop other drivers and local network protocol routines.

9.2.2 Multiple Internet Addresses and Gateway Forwarding

The multiple interface extension necessitated handling a host having more than one internet address. Also, gateway forwarding was added.

As described above, there is a table of local host addresses that may be associated with one or more physical network interfaces. When a connection is opened, the user may optionally specify a source address for it. If the source address is specified, a search is made of the local address table, and that address is used subsequently if it is found. An error results if it is not found. If the source address is unspecified, a check is first made to see if the destination is on any of the networks that the local host has interfaces on. If so, that local address is used. Otherwise, a check of the gateway table is made to determine if there is a gateway to the network of the destination on any of the local host's networks. Barring that, the first entry in the local address table is used as the default.

When a message is sent out over a TCP/IP connection, the IP output routine determines a local network address (of the destination host or a gateway to it) and an interface on which to

send the message. If the destination host is on a different network from the source, the gateway table is searched for a gateway to the destination. The gateway table consists of entries with source and destination network addresses and the corresponding local network address of the gateway. If no direct gateway is found between the source and destination networks, an alternative "smart" gateway is used. Once a gateway is found, the address is saved in the per connection control block. ICMP redirect messages are used to update this information, if necessary.

The gateway table is initialized at boot time from a file in the UNIX file system. This file is in binary format and is generated from an ASCII file with a conversion program. A UNIX ioctl call allows the gateway file to be reread at any time.

9.2.3 Raw User Interface

The raw IP and local network interfaces have been extended to allow for more flexibility in building packet headers and demultiplexing incoming messages. The user can open network "connections" over which he will send and/or receive raw IP or local network protocol packets.

To receive packets, the user can specify a range of link or protocol numbers, and optionally foreign and local network addresses. The ranges may not overlap. If the foreign address is specified, only packets from that address will be returned. Otherwise, packets with the specified link or protocol number from all foreign hosts will be received. Similarly, if the local address is specified, only packets sent to that address will be returned. Read requests on raw connections return at most one packet (or less if a read length smaller than the packet size was specified). The entire packet, including the header, is returned. An option is available to enable reception of ICMP or local network protocol error and control messages for the specified protocol number, as well as normal data packets. There is a buffer limit for packets being received. If it is exceeded. newly received packets will be dropped.

For sending packets, the user can specify if he wants the system to construct the IP or local network protocol headers, or if he wants to supply the headers and have the packets sent "as is." There is a third option for raw IP connections that causes the checksum field of a user-supplied header to be filled in by the system. If the system supplies the headers, the link/protocol numbers and net addresses are taken from the parameters specified when the "connection" is opened. Raw IP messages go through the same local route processing, described

above, as TCP/IP messages do. For packets sent "as is," no local network processing (such as RFNM counting for ARPANET 1822 protocol) is done. There is a buffer limit for packets being sent. If it is exceeded, the packet will not be accepted from the user, and the send call will not block. If the packet cannot be sent from the host, an error code will be returned to the user.

9.2.4 ICMP Messages

The ICMP messages (formerly Gateway-Host GGP messages) are now being handled. Specifically, redirect and destination unreachable messages affect the IP output routing and gateway forwarding procedures described above. It is unclear what to do with source quench messages, and it is expected that future work will be done on the appropriate response to this type of message and to the issue of internet flow control in general. The extensions to ICMP that were recently adopted by the Internet Working Group (addition of checksums and changes in message formats) will be implemented when those changes go into effect.

9.3 Future Work

Development and debugging of a new version of TCP/IP, with the extensions described above, will continue during the coming quarter. In addition, debugging will continue on the current version and the user protocols (TELNET, FTP, and MTP), with testing between BBN and the beta test sites. A general release of the TCP/IP and higher level protocol software is expected by November.

DISTRIBUTION

ARPA
Director (3 copies)
Defense Advanced Research Projects Agency
1400 Wilson Blvd.
Arlington, VA 22209
Attn: Program Manager

R. Kahn
V. Cerf
R. Ohlander
D. Adams

DEFENSE DOCUMENTATION CENTER (12 copies)
Cameron Station
Alexandria, VA 22314

DEFENSE COMMUNICATIONS ENGINEERING CENTER
1850 Wiehle Road
Reston, VA 22090
Attn: Lt. Col. F. Zimmerman

DEPARTMENT OF DEFENSE
9800 Savage Road
Ft. Meade, MD 20755
R. McFarland R17 (2 copies)
M. Tinto S46 (2 copies)

DEFENSE COMMUNICATIONS AGENCY
8th and South Courthouse Road
Arlington, VA 22204
Attn: Code 252

NAVAL ELECTRONIC SYSTEMS COMMAND
Department of the Navy
Washington, DC 20360
B. Hughes, Code 6111
F. Deckelman, Code 6131
J. Machado, Code 6134

BOLT BERANEK AND NEWMAN INC.
1701 North Fort Myer Drive
Arlington, VA 22209
E. Wolf

DISTRIBUTION cont'd

BOLT BERANEK AND NEWMAN INC.
50 Moulton Street
Cambridge, MA 02138

G. Falk
R. Bressler
A. Lake
J. Robinson
A. McKenzie
F. Heart
P. Santos
R. Brooks
W. Edmond
J. Haverty
D. McNeill
M. Brescia
A. Nemeth
B. Woznick
R. Thomas
W. Milliken
S. Groff
M. Hoffman
R. Rettberg
W. Mann
P. Carvey
D. Hunt
P. Cudhea
L. Evenchik
D. Flood Page
J. Herman
J. Sax
R. Hinden
G. Ruth
S. Kent
R. Gurwitz
E. Starr
A. Sheltzer
Library